In the Claims:

Please amend claims 1, 8, 10, 11 and 51 as indicated below.

1. (Currently amended) A method, comprising:

an application sending to a container a data object specifying a high-level function provided by a system external to the application;

the container receiving the data object specifying the high-level function;

the container accessing metadata corresponding to the high-level function ~~call~~, wherein the metadata describes the high-level function of the external system;

the container performing one or more transformations on the data object specifying the high-level function in accordance with the metadata to produce a data object including information for driving a connector to the external system to make a plurality of low-level calls to the external system to perform the high-level function;

the container executing the data object including the information for driving the connector to the external system to drive the connector to make the plurality of low-level calls to the external system; and

the external system executing a plurality of low-level functions as specified by the plurality of low-level calls received from the connector, wherein said executing the plurality of low-level functions performs the high-level function of the external system ~~specified by the high-level function call~~.

2. (Original) The method as recited in claim 1, further comprising:

the container receiving results of said executing the plurality of low-level functions;

the container storing the results of said executing the plurality of low-level functions in a results data object;

the container performing one or more transformations on the results data object to generate an output data object, wherein the output data object includes the results of the high-level function; and

the container providing the output data object to the application.

3. (Original) The method as recited in claim 1, wherein the metadata is accessed from a metadata repository.

4. (Original) The method as recited in claim 3, wherein the metadata repository is in a persistent store that provides access to the metadata repository to applications within managed environments and applications within unmanaged environments.

5. (Original) The method as recited in claim 4, wherein the application is in an unmanaged environment, the method further comprising the application moving from the unmanaged environment to a managed environment, wherein said moving the application does not require repopulating the metadata repository.

6. (Original) The method as recited in claim 3, wherein the metadata repository is implemented in a Java Naming and Directory Interface (JNDI) namespace.

7. (Original) The method as recited in claim 1, further comprising, prior to sending the data object specifying the high-level function, the application accessing a metadata repository comprising information describing one or more high-level functions

of the external system to discover the one or more high-level functions provided by the external system, wherein the specified high-level function is one of the one or more discovered high-level functions.

8. (Currently amended) The method as recited in claim 1, further comprising, prior to the application sending the data object specifying the high-level function:

accessing a metadata repository comprising information describing one or more functions of the external system; and

generating the metadata corresponding to the high-level function ~~call~~ from the information describing the one or more functions of the external system.

9. (Original) The method as recited in claim 1, wherein the application accesses the container in accordance with an Application Programming Interface (API) to the container.

10. (Currently amended) The method as recited in claim 1, wherein the plurality of low-level function calls to the external system are specific to the connector, wherein said mapping the high-level function ~~call~~ to a plurality of low-level function calls to the external system comprises the container accessing metadata corresponding to the high-level function ~~call~~, wherein the metadata maps the high-level function ~~call~~ to the plurality of low-level function calls to the external system specific to the connector.

11. (Currently amended) The method as recited in claim 10, further comprising modifying the metadata corresponding to the high-level function to map the high-level function ~~call~~ to a plurality of low-level function calls to the external system specific to a different connector to the external system.

12. (Original) The method as recited in claim 1, wherein the application is comprised in the container.

13. (Original) The method as recited in claim 1, wherein the application is external to the container.

14. (Original) The method as recited in claim 1, wherein the connector is a Java 2 Enterprise Edition Connector Architecture (J2EE CA) connector.

15. (Original) The method as recited in claim 1, wherein the container is an application server.

16. (Original) The method as recited in claim 1, wherein the container is a Java Virtual Machine (JVM).

17. (Original) The method as recited in claim 1, wherein the external system is an Enterprise Information System (EIS).

18. (Original) A method comprising:

an application sending to a container a high-level function call corresponding to a high-level function of a system external to the application;

the container mapping the high-level function call to a series of low-level function calls to the external system;

the container driving a connector to the external system to make the series of low-level function calls to the external system; and

the external system executing the series of low-level functions as specified by the series of low-level function calls received from the connector, wherein said executing the series of low-level functions performs the high-level

function of the external system corresponding to the high-level function call.

19. (Original) The method as recited in claim 18, wherein said mapping the high-level function call to a series of low-level function calls to the external system comprises:

the container accessing metadata corresponding to the high-level function call, wherein the metadata describes the high-level function of the external system, wherein the data is accessed from a metadata repository comprising metadata describing a plurality of high-level functions of the external system; and

the container performing one or more transformations on the high-level function call in accordance with the metadata to produce a data object, wherein the data object includes information for driving a connector to the EIS to make the series of low-level calls to the external system to perform the high-level function.

20. (Original) The method as recited in claim 18, wherein, prior to sending the high-level function call, the method further comprises the application accessing a metadata repository comprising metadata describing a plurality of high-level functions of the external system to discover one or more high-level functions provided by the external system, wherein the specified high-level function is one of the one or more discovered high-level functions.

21. (Original) The method as recited in claim 20, wherein the metadata repository is in a persistent store that provides access to the metadata repository to applications within managed environments and applications within unmanaged environments.

22. (Original) The method as recited in claim 20, wherein the application is in an unmanaged environment, the method further comprising the application moving from the

unmanaged environment to a managed environment, wherein said moving the application does not require repopulating the metadata repository.

23. (Original) The method as recited in claim 20, wherein the metadata repository is implemented in a Java Naming and Directory Interface (JNDI) namespace.

24. (Original) The method as recited in claim 18, further comprising:

the container receiving results of said executing the plurality of low-level functions;

the container storing the results of said executing the plurality of low-level functions in a results data object;

the container performing one or more transformations on the results data object to generate an output data object, wherein the output data object includes the results of the high-level function; and

the container providing the output data object to the application.

25. (Original) The method as recited in claim 18, further comprising, prior to sending the high-level function call:

accessing a metadata repository comprising information describing one or more functions of the external system; and

generating a metadata description of the high-level function from the information describing the one or more functions of the external system;

wherein said mapping the high-level function call to a series of low-level function calls to the external system is performed in accordance with the metadata description of the high-level function.

26. (Original) The method as recited in claim 18, wherein the application accesses the container in accordance with an Application Programming Interface (API) to the container.

27. (Original) The method as recited in claim 18, wherein the container accesses the connector in accordance with a connector Application Programming Interface (API) to the connector, wherein the connector API defines each of the series of low-level function calls to the external system through the connector, wherein said mapping the high-level function call to a series of low-level function calls to the external system comprises the container accessing metadata corresponding to the high-level function call, wherein the metadata maps the high-level function call to the series of low-level function calls to the external system as defined by the connector API.

28. (Original) The method as recited in claim 27, further comprising modifying the metadata corresponding to the high-level function to map the high-level function call to a series of low-level function calls to the external system as defined by a connector API corresponding to a different connector to the external system.

29. (Original) The method as recited in claim 28, further comprising:

the application resending to the container the high-level function call corresponding to the high-level function of the external system;

the container accessing the metadata corresponding to the high-level function call to map the high-level function call to the series of low-level function calls to the external system as defined by the connector API to the different connector to the external system;

the container driving the different connector to the external system to make the series of low-level function calls to the external system as defined by the connector API to the different connector to the external system; and

the external system executing the series of low-level functions as specified by the series of low-level function calls received from the different connector, wherein said executing the series of low-level functions performs the high-level function of the external system corresponding to the high-level function call.

30. (Original) The method as recited in claim 18, wherein the connector is a Java 2 Enterprise Edition Connector Architecture (J2EE CA) connector.

31. (Original) The method as recited in claim 18, wherein the container is an application server.

32. (Original) The method as recited in claim 18, wherein the container is a Java Virtual Machine (JVM).

33. (Original) The method as recited in claim 18, wherein the external system is an Enterprise Information System (EIS).

34. (Original) The method as recited in claim 18, wherein the application is loosely coupled to the container.

35. (Original) The method as recited in claim 18, wherein the application is tightly coupled to the container.

36. (Withdrawn) A system comprising:

an Enterprise Information System (EIS);

an application;

a container, wherein the EIS is external to the container, and wherein the container comprises:

a connector, wherein the connector is configured to provide an interface between low-level functions of the EIS and the application;

an adapter configured to:

receive a high-level function call from the application, wherein the high-level function call specifies a high-level function of the EIS;

map the high-level function call to a series of low-level function calls to the EIS; and

drive the connector to make the series of low-level function calls to the EIS;

wherein the EIS is configured to execute the series of low-level functions in response to the series of low-level function calls, wherein said executing the series of low-level functions performs the high-level function of the EIS specified by the high-level function call.

37. (Withdrawn) The system as recited in claim 36, wherein, in said mapping the high-level function call to a series of low-level function calls to the EIS, the adapter is further configured to:

access metadata corresponding to the high-level function call, wherein the metadata describes the high-level function of the EIS; and

perform one or more transformations on the high-level function call in accordance with the metadata to produce a data object, wherein the data object includes information for driving a connector to the EIS to make a plurality of low-level calls to the EIS to perform the high-level function.

38. (Withdrawn) The system as recited in claim 37, further comprising:

a metadata repository comprising metadata describing a plurality of high-level functions of the EIS;

wherein, in said accessing metadata corresponding to the high-level function call, the adapter is further configured to access the metadata from the metadata repository.

39. (Withdrawn) The system as recited in claim 38, wherein the metadata repository is in a persistent store that provides access to the metadata repository to applications within managed environments and applications within unmanaged environments.

40. (Withdrawn) The system as recited in claim 38, wherein the application is in an unmanaged environment, wherein the application is configured to move from the unmanaged environment to a managed environment without repopulating the metadata repository.

41. (Withdrawn) The system as recited in claim 38, wherein the metadata repository is implemented in a Java Naming and Directory Interface (JNDI) namespace.

42. (Withdrawn) The system as recited in claim 36, further comprising:

a metadata repository comprising metadata describing a plurality of high-level functions of the EIS;

wherein, prior to sending the high-level function call, the application is configured to access the metadata repository to discover one or more high-level functions provided by the EIS from the metadata describing the plurality of high-level functions, wherein the specified high-level function is one of the one or more discovered high-level functions.

43. (Withdrawn) The system as recited in claim 36, wherein the adapter is further configured to:

receive results of said executing the plurality of low-level functions;

store the results of said executing the plurality of low-level functions in a results data object;

perform one or more transformations on the results data object to generate an output data object; and

provide the output data object to the application, wherein the output data object includes the results of the high-level function.

44. (Withdrawn) The system as recited in claim 36, further comprising:

a metadata repository comprising metadata describing a plurality of high-level functions of the EIS;

wherein the application is further configured to:

access the metadata repository; and

generate a metadata description of the high-level function from the information describing the one or more functions of the EIS;

wherein, in said mapping the high-level function call to a series of low-level function calls to the EIS, the adapter is further configured to:

access the metadata description of the high-level function; and

perform said mapping in accordance with the metadata description.

45. (Withdrawn) The system as recited in claim 36, further comprising an Application Programming Interface (API) for the adapter, wherein the application is configured to access the adapter in accordance with the adapter API.

46. (Withdrawn) The system as recited in claim 36, wherein the system further comprises:

an Application Programming Interface (API) to the connector that defines each of the series of low-level function calls to the EIS through the connector, wherein the adapter is further configured to access the connector in accordance with the connector API; and

a metadata repository comprising a metadata description of the high-level function of the EIS, wherein the metadata description maps the high-level function call to the series of low-level function calls to the EIS as defined by the connector API;

wherein, in said mapping the high-level function call to a series of low-level function calls to the EIS, the adapter is further configured to access the

metadata description of the high-level function call comprised in the metadata repository.

47. (Withdrawn) The system as recited in claim 46, wherein the container further comprises:

a different connector configured to provide a different interface between the low-level functions of the EIS and the application; and

an API to the different connector that defines each of the series of low-level function calls to the EIS through the different connector, wherein the adapter is further configured to access the different connector in accordance with the API to the different connector;

wherein the metadata description of the high-level function call comprised in the metadata repository is modifiable to map the high-level function call to the plurality of low-level function calls to the EIS as defined by the API to the different connector.

48. (Withdrawn) The system as recited in claim 36, wherein the connector is a Java 2 Enterprise Edition Connector Architecture (J2EE CA) connector.

49. (Withdrawn) The system as recited in claim 36, wherein the container is an application server.

50. (Withdrawn) The system as recited in claim 36, wherein the container is a Java Virtual Machine (JVM).

51. (Currently amended) A system, comprising:

an Enterprise Information System (EIS);

an application;

a container, wherein the EIS is external to the container, and wherein the container comprises:

a connector, wherein the connector is configured to provide an interface between low-level functions of the EIS and the application;

means for receiving a high-level function call from the application;

means for mapping the high-level function call to a series of low-level function calls to the EIS; and

means for driving the connector to make the series of low-level function calls to the EIS;

wherein the EIS is configured to execute the series of low-level functions in response to the series of low-level function calls, wherein said executing the series of low-level functions performs ~~the~~ a high-level function of the EIS corresponding to the high-level function call.

52. (Original) The system as recited in claim 51, wherein, in said mapping the high-level function call to a series of low-level function calls to the EIS, the container further comprises means for:

accessing metadata corresponding to the high-level function call, wherein the metadata describes the high-level function of the EIS; and

performing one or more transformations on the high-level function call in accordance with the metadata to produce a data object, wherein the data

object includes information for driving a connector to the EIS to make a plurality of low-level calls to the EIS to perform the high-level function.

53. (Original) The system as recited in claim 52, further comprising a metadata repository comprising metadata describing a plurality of high-level functions of the EIS, wherein the metadata is accessed from the metadata repository.

54. (Original) The system as recited in claim 53, further comprising means for providing access to the metadata repository to applications within managed environments and applications within unmanaged environments.

55. (Original) The system as recited in claim 53, wherein the application is in an unmanaged environment, further comprising means to move the application from the unmanaged environment to a managed environment without repopulating the metadata repository.

56. (Original) The system as recited in claim 53, wherein the metadata repository is implemented in a Java Naming and Directory Interface (JNDI) namespace.

57. (Original) The system as recited in claim 51, wherein the container further comprises means for:

receiving results of said executing the plurality of low-level functions;

storing the results of said executing the plurality of low-level functions in a results data object;

performing one or more transformations on the results data object to generate an output data object; and

providing the output data object to the application, wherein the output data object includes the results of the high-level function.

58. (Original) The system as recited in claim 51, wherein the container further comprises:

a different connector, wherein the different connector is configured to provide an interface between the low-level functions of the EIS and the application; and

means for driving the different connector to make the series of low-level function calls to the EIS.

59. (Original) The system as recited in claim 51, wherein the connector is a Java 2 Enterprise Edition Connector Architecture (J2EE CA) connector.

60. (Original) The system as recited in claim 51, wherein the container is an application server.

61. (Original) The system as recited in claim 51, wherein the container is a Java Virtual Machine (JVM).

62. (Withdrawn) A metadata-aware adapter for a connector to an Enterprise Information System (EIS), configured to:

receive a high-level function call for a high-level functions of the EIS;

map the high-level function call to a series of low-level function calls to the EIS; and

drive the connector to make the series of low-level function calls to the EIS;

wherein the EIS is configured to execute the series of low-level functions in response to the series of low-level function calls, wherein said executing the series of low-level functions performs the high-level function of the EIS.

63. (Withdrawn) The metadata-aware adapter as recited in claim 62, wherein, in said mapping the high-level function call to a series of low-level function calls to the EIS, the metadata-aware adapter is further configured to:

access metadata corresponding to the high-level function call, wherein the metadata describes the high-level function of the EIS; and

perform one or more transformations on the high-level function call in accordance with the metadata to produce a data object, wherein the data object includes information for driving the connector to the EIS to make a plurality of low-level calls to the EIS to perform the high-level function.

64. (Withdrawn) The metadata-aware adapter as recited in claim 63, wherein, in said accessing metadata corresponding to the high-level function call, the metadata-aware adapter is further configured to access the metadata from a metadata repository comprising metadata describing a plurality of high-level functions of the EIS.

65. (Withdrawn) The metadata-aware adapter as recited in claim 64, wherein the metadata repository is in a persistent store.

66. (Withdrawn) The metadata-aware adapter as recited in claim 62, further configured to:

receive results of said executing the plurality of low-level functions;

store the results of said executing the plurality of low-level functions in a results data object;

perform one or more transformations on the results data object to generate an output data object; and

provide the output data object to an application from which the high-level function call was received, wherein the output data object includes the results of the high-level function.

67. (Withdrawn) The metadata-aware adapter as recited in claim 62, wherein the metadata-aware adapter and the connector are comprised in a container, and wherein the high-level function call was generated by an application external to the container.

68. (Withdrawn) The metadata-aware adapter as recited in claim 67, further comprising an Application Programming Interface (API) for the adapter, wherein the application is configured to access the adapter in accordance with the adapter API.

69. (Withdrawn) The metadata-aware adapter as recited in claim 62, wherein the series of low-level function calls to the EIS are specific to the connector, wherein, in said mapping the high-level function call to a series of low-level function calls to the EIS, the adapter is further configured to access metadata corresponding to the high-level function call, wherein the metadata maps the high-level function call to the series of low-level function calls to the EIS specific to the connector.

70. (Withdrawn) The metadata-aware adapter as recited in claim 69, wherein the metadata corresponding to the high-level function is modifiable to map the high-level function call to a series of low-level function calls to the EIS specific to a different connector to the EIS.

71. (Withdrawn) The metadata-aware adapter as recited in claim 62, wherein the connector is a Java 2 Enterprise Edition Connector Architecture (J2EE CA) connector.

72. (Withdrawn) The metadata-aware adapter of claim 62, further configured to install in a container comprising the connector.

73. (Previously presented) A tangible computer readable medium comprising program instructions, wherein the program instructions are computer-executable to implement:

an application sending to a container a high-level function call corresponding to a high-level function of a system external to the application;

the container mapping the high-level function call to a series of low-level function calls to the external system;

the container causing a connector to the external system comprised in the container to make the series of low-level function calls to the external system; and

the external system executing the series of low-level functions as specified by the series of low-level function calls received from the connector, wherein said executing the series of low-level functions performs the high-level function of the external system corresponding to the high-level function call.

74. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the program instructions are further computer-executable to implement:

the container receiving results of said executing the plurality of low-level functions;

the container storing the results of said executing the plurality of low-level functions in a results data object;

the container performing one or more transformations on the results data object to generate an output data object, wherein the output data object includes the results of the high-level function; and

the container providing the output data object to the application.

75. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the program instructions are further computer-executable to implement including metadata describing a plurality of high-level functions of the external system in a metadata repository.

76. (Previously presented) The tangible computer readable medium as recited in claim 75, wherein the program instructions are further computer-executable to implement, prior to sending the high-level function call, the application accessing the metadata repository to discover one or more high-level functions provided by the external system from the metadata describing the plurality of high-level functions, wherein the specified high-level function is one of the one or more discovered high-level functions.

77. (Previously presented) The tangible computer readable medium as recited in claim 75, wherein the metadata repository is in a persistent store, wherein the application is in an unmanaged environment, wherein the program instructions are further computer-executable to implement the application moving from the unmanaged environment to a managed environment without repopulating the metadata repository.

78. (Previously presented) The tangible computer readable medium as recited in claim 75, wherein the metadata repository is in a Java Naming and Directory Interface (JNDI) namespace.

79. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the application accesses the container in accordance with an Application Programming Interface (API) to the container.

80. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the container accesses the connector in accordance with a connector Application Programming Interface (API) to the connector, wherein the connector API defines each of the series of low-level function calls to the external system through the connector, wherein said mapping the high-level function call to a series of low-level function calls to the external system comprises the container accessing metadata corresponding to the high-level function call, wherein the metadata maps the high-level function call to the series of low-level function calls to the external system as defined by the connector API.

81. (Previously presented) The tangible computer readable medium as recited in claim 80, wherein the program instructions are further computer-executable to implement modifying the metadata corresponding to the high-level function to map the high-level function call to the series of low-level function calls to the external system as defined by a connector API corresponding to a different connector to the external system.

82. (Previously presented) The tangible computer readable medium as recited in claim 81, wherein the program instructions are further computer-executable to implement:

the application resending to the container the high-level function call corresponding to the high-level function of the external system;

the container accessing the metadata corresponding to the high-level function call to map the high-level function call to the series of low-level function calls to the external system as defined by the connector API to the different connector to the external system;

the container driving the different connector to the external system to make the series of low-level function calls to the external system as defined by the connector API to the different connector to the external system; and

the external system executing the series of low-level functions in response to the series of low-level function calls made by the different connector to the external system, wherein said executing the series of low-level functions performs the high-level function of the external system.

83. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the connector is a Java 2 Enterprise Edition Connector Architecture (J2EE CA) connector.

84. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the container is an application server.

85. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the container is a Java Virtual Machine (JVM).

86. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the external system is an Enterprise Information System (EIS).

87. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the application is loosely coupled to the container.

88. (Previously presented) The tangible computer readable medium as recited in claim 73, wherein the application is tightly coupled to the container.